2

USAISEC

US Army Information Systems Engineering Command
Fort Huachuca, AZ 85613-5300

U.S. ARMY INSTITUTE FOR RESEARCH
IN MANAGEMENT INFORMATION,
COMMUNICATIONS, AND COMPUTER SCIENCES
(AIRMICS)

GENERIC DISTRIBUTED SYSTEMS MODEL

(ASQBG-C-89-024)

March, 1989

AIRMICS
115 O'Keefe Building
Georgia Institute of Technology
Atlanta, GA 30332-0800

Generic Distributed System Model

by

Dr. Dharma P. Agrawal

Department of Electrical & Computer Engineering
Box 7911
North Carolina State University
Raleigh, NC 27695-7911

for

Army Institute for Research in Management Information,
Communications, and Computer Science (AIRMICS),
115 O'Keefe Building
Georgia Institute of Technology
Atlanta, Georgia 30332-0800

May 4, 1988

Contract No. DAAL03-86-D-001
Delivery Order 0757
Scientific Services Program

Preface

This report has been prepared for the AIRMICS located in the heart of Atlanta. I am thankful to Dr. C. Ronald Green, Chief of the Communication and Network System Division and Mr. John Mitchell, the Director of the AIRMICS, for giving me this opportunity to work on this project. I thoroughly enjoyed my stay at AIRMICS, specially interacting with the friendly engineers. My sincere thanks are also due to my family members for their patience and encouragement.

GENERIC DISTRIBUTED SYSTEM MODEL

Summary

This report identifies various issues associated with the design of Distributed Systems and presents an overview of modeling. It is felt that the tools are primarily oriented towards development of software to run on distributed systems rather than the design tools for distributed systems. During my four months' of stay at AIRMICS, information about some design tools were acquired. But additional efforts are needed to find out other modeling efforts being pursued by various other funding agencies and research organizations so that any replication of efforts could be avoided and the problem of Distributed System Design for a given requirements and specifications could be addressed effectively. Future directions and recommendations have also been included as a part of this report.

Introduction

The term distributed system applies to a very generic class of decentralized systems. It encompasses all systems having geographically distributed multiple processors with some kind of coordination between them. Usually, people confuse a simple networking of microcomputers or work-stations with a distributed system and a clear distinction between the two needs to be made. What is expected in a distributed system is an appropriate coordination of data and control among various functional units. In a networked system, all the computers are autonomous and each one works independently of others, even though messages could be transferred between them. The name distributed system also implies that the resources (including computers) are dispersed physically, with one or many locations acting as the source of data and others taking care of processing the data and making decisions. The area of distributed processing is relatively new and increased emphasis is needed to obtain major results. There are numerous open questions and all of these need to be addressed carefully to provide results of substantial importance. The complication in such efforts arises from various associated factors and some of them are as follows:

i. How many and what type of resources are required in a heterogeneous environment?

ii. What type of data is anticipated?

iii. What are the user defined requirements that could adequately represent the real-life situation?

iv. What is the best way of translating users' requirements into system level requirements?

v. What are the appropriate ways of system modeling?

vi. What techniques ought to be used in partitioning the system model?

vii. What are the target system architectures and network topologies?

viii. What are the ways to incorporate the multi-media networks?

ix. How to take into account the reliability, graceful degradation and recovery aspects concurrently?

x. What strategies ought to be used in allocating files and system software to various nodes of the system?

xi. What alternate nodal design ought to be considered?

xii. What are the essential performance parameters?

It may be noted that the distributed system issues have been addressed in a piecemeal manner and much attention have been paid to the modeling of software for distributed system applications. There has been very little work on the modeling of the overall distributed system and an overview of existing work is included in the next section.

Overview of Existing Work

The design of a distributed system for a given class of applications is a very complex problem and has been of constant interest to the research and development community. Some of the analysis techniques have been covered in [CHA80] while an overview of distributed computer systems has been given in [STA84]. Gonzalez and Jordan [GON80] were the first ones to come up with the framework for quantitative evaluation of distributed computer systems and some details are given in the next section [MCA87]. The four major areas covered in the literature are the network architectures, the resource allocation, the software design issues and the language selection. Networks [PRA85] suitable for both parallel and distributed processing have been introduced in the literature. A comparative study [AGR85] of many such networks have been performed in terms of various architectural characteristics such as average distance, maximum distance (diameter) number of alternate paths, ease of expansion, simplicity of routing, and number of input/output ports per node. The effectiveness of these network configurations from an algorithmic point of view has also been determined [AGR85] when some of the actual algorithms are mapped onto various architectures. Infact, this comparative study [AGR85] led to the development of the B-HIVE [AGR88], a 24-node multicomputer project at the North Carolina State University which is nearing completition and will be used for parallel and distributed processing experimentation.

The resource allocation (sometimes also called mapping in the context of parallel processing) problem has been addressed by various researchers [PAT84, COR87, KAL87, TAN85, WAN85, YAS87, RAM83, GAI87, HOU87]. The basic idea in all these efforts is to have models of the applications and allocate various parts of the models to different processors or nodes of the system. The allocation is done either statically or pseudo-dynamically and the usual performance parameters to be optimized are the turn-around time, the speed up, the processor utilization and the channel utilization. These parameters are similar when applied to a distributed system and a parallel system. Most of the researchers consider all the tasks independent of others and are usually allocated to reduce the net turn-around time. For parallel system, the precedence relationship (or dependence) is also taken in to account while doing the mapping. The dependency relations are incorporated in some distributed systems [CAT87] in the form of data-flow graphs when employed for real-time applications. This needs to be done for any application as long as some coordination is desirable and in future, there will be increased emphasis on this aspect. Recent results on allocation [LEU87, CAS88, CHI88] emphasize the need for minimization of communication overhead by either overlapping the communication with computation or appropriately managing the resources. The static mapping with no run-time overhead, is an approximation of the actual requirements; while the dynamic scheduling done at run time, in general, requires prohibitively large overhead. A good compromise is yet to be found. In a similar way, there are many open questions and much more research is needed in this area.

The software issues have been the bottleneck even for a uniprocessor system and software modeling for a distributed system has been a lively subject and poses new challanges. Some of the noteworthy approaches include [SHA87, IND86, CHE80, RAM85] while the recent spiral model [BOE86] advocates the adoption of successive enhancements untill the specifications are satisfied. In terms of actual coding, relative advantages and disadvantages of various languages have been described in [FOR86]. The use of communicating sequential processes for distributed system software has been demonstrated in [HAN87, RAV87]. The AIRMICS is in process of acquiring the Joyce environment from Syracuse University and this could prove to be very useful. It may be noted that all these efforts have been directed towards development of efficient software to run on a distributed system, rather than a tool for distributed system design. In January 1988, when I joined AIRMICS, the only design tool available in-house was the McCable Software package. It took a while to set up an appropriate system configuration so that its capabilities could be tested. During my four months' stay at AIRMICS, I was able to find out more about some other design tool efforts being pursued. In April 1988, we were able to acquire the executable code for the TRW DCDS software package from the US Army Strategic Defense Command at Huntsville. But, because of numerous reasons, we were not able to install this DCDS package neither at AIRMICS nor at the Georgia Tech Computing Center.

Other modeling efforts that we came accross have been associated with the USA ISEC Information Software Support Command (ISSC) Quality Assurance Directorate (QAD) and the USAF Rome Air Development Center (RADC). The RADC work (RAD86) written in SYMSCRIPT, has been basically applicable for the network modeling and will be released in July 1989 after appropriate update. The QAD has been supporting queueing delay computation work for IBM machines and is currently seeking help in modeling Sperry 5000 systems. We tried our best to get their reports, but as of this writing, we have not received any documents.

McCabe Design Tool

The McCabe Design Tool [MCC87] has been developed under the sponsorship of AIRMICS. The basic idea is to have a tool for distributed system design with a capability of varying different parameters and observing their impact on various quality attributes such as:

-Availability,
-Efficiency,
-Flexibility,
-Integrity,
-Maintainability,
-Reliability,
-Survivability.

These parameters are good to obtain a reasonable idea of the design adequacies. But a careful execution of the software would reveal that the designer is expected to provide a lot of information while no optimization technique has been used. For example, the designer needs to specify all the requirements as follows (Fig. 1):

-The no. of external request for processing per day.
-The physical distance between various places where queries are generated.
-The type of processing required in terms of function (or process).
-The time required by each process.
-The probability of one process utilizing another process.
-Placement restrictions on some external requests or for some processes.
-Any other restrictions or requirements such as response time, etc.

Once these have been supplied in the form of the model (see Fig. 2), and the probable physical locations of each process have been given, then the software package computes various attributes (shown in Table I and Table II). At this point, it is possible to change the location of each process, the distance between the query sources, or the weights assigned to various attributes and the impact on the net system performance could be observed. In this way, a lot of information has to be provided by the designer and there is no provision for parameter optimization. All changes have to be done manually and no decision could be made about the type of processing nodes to be used or the type of channels and their capacity to be employed. But, it could be said to be a mere starting point and substantial efforts are needed to make it comprehensive for today's technology.

INFORMATION PERTAINING TO LOCATIONS AND POLICY CAN BE COMBINED WITH THE INITIAL DIAGRAM TO PRODUCE A PARTITIONED DFD. THE BOLD LINES REPRESENT SERVICES WHICH WERE PLACED BECAUSE OF MANDATORY POLICY CONSTRAINTS. THE REMAINING PROCESSES WERE LOCATED ARBITRARILY FOR THIS BASELINE ALLOCATION.

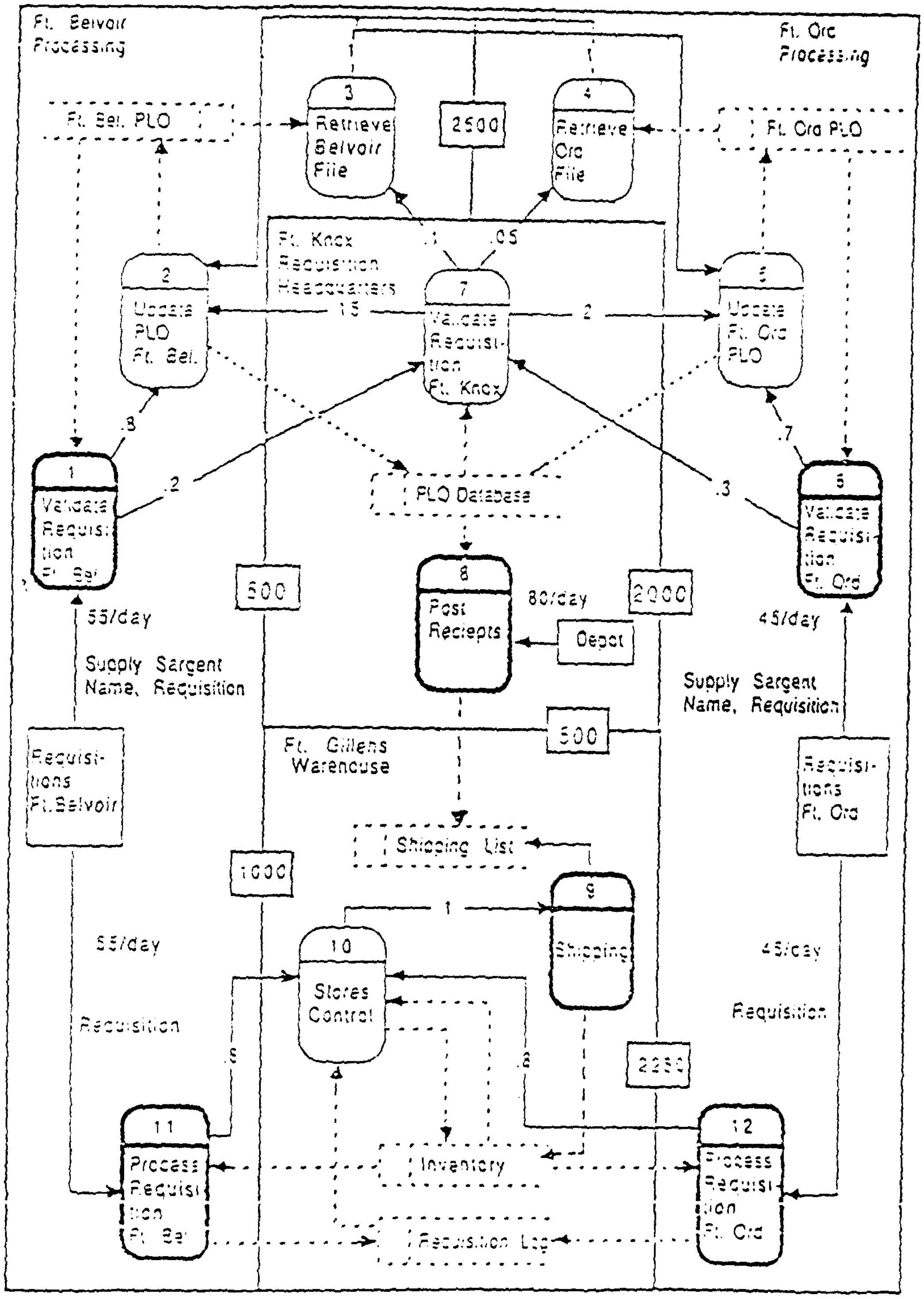


Fig. 1.POLICY DICTATES THAT THE APPLICATION HEADQUARTERS WILL MAINTAIN THE PRESCRIBED LOAD LIST (PLO) AND THE POST RECEIPTS DEPARTMENT . IN ADDITION, REQUISITION SITES MUST BE LOCATED AT FT. BELVOIR AND FT. ORD SO THAT REQUISITIONS CAN BE MADE DIRECTLY.

COMPLETE DFD CREATED USING DESIGN AID, APPEARS BELOW. THE DIAGRAM IS APPENDED WITH THE SPECIAL SYMBOLS WHICH CONTAIN THE SYSTEM PARAMETERS: LOCATION, NAME, UTILIZATION, EXTERNAL ENTRIES, DATA FLOW PROBABILITIES.

s: service no. & name

e: external entry demand placed

u: utilization & service name

l: service location

p: probability of connections between pair of services

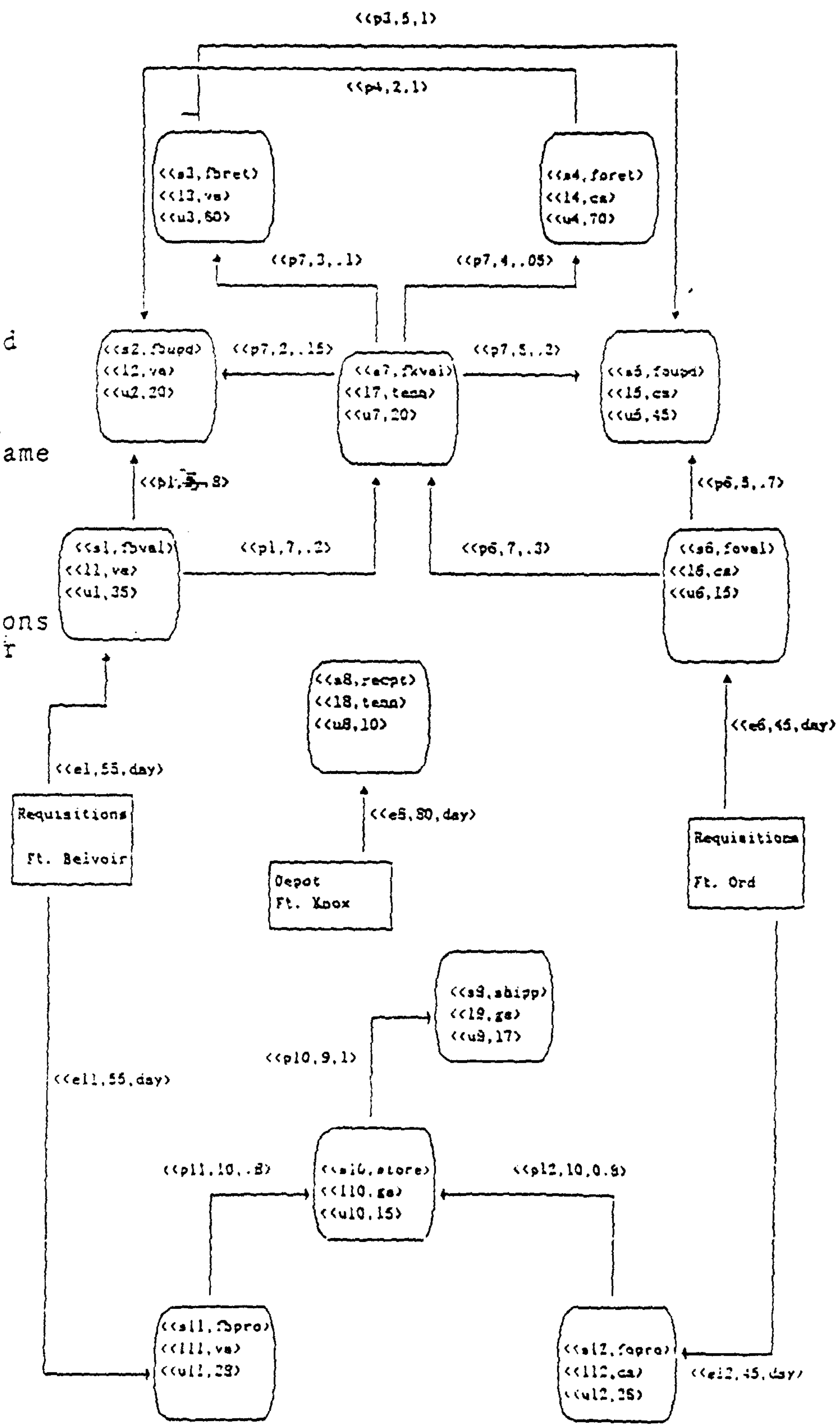


Fig. 2. Application Model

Table I. PERFORMANCE ANALYSIS RESULTS

SUMMARY OF PERFORMANCE ANALYSIS

Serv. Name	% Util.	Ext. Ent. / DAY	Serv. Rate / DAY	Exp. Work-load/ DAY	% Contribution To PT	% Contribution To IT
fbval	35	55.00	157.14	55.00	9.43	5.23
fbupd	20	0.00	244.50	48.90	5.39	1.39
fbret	60	0.00	4.08	2.45	16.17	25.00
foret	70	0.00	1.75	1.22	18.87	45.37
foupd	45	0.00	86.33	38.85	12.13	10.23
foval	15	45.00	300.00	45.00	4.04	0.74
fkval	20	0.00	122.50	24.50	5.39	1.39
recpt	10	80.00	800.00	80.00	2.70	0.31
shipp	17	0.00	533.33	80.00	4.04	0.74
store	15	0.00	470.59	80.00	4.58	0.97
fbpro	28	55.00	196.43	55.00	7.55	3.02
fopro	36	45.00	125.00	45.00	9.70	5.62

End-to-End Time (ETE): 7.31 DAY

% ETE Processing Time (PT): 50.75 Idle Time (IT):49.25

Table II. COST ANALYSIS RESULTS

SUMMARY OF COST ANALYSIS

Service Name	Service Location	Number Of Outflows	Ext. Ent. /DAY	Exp. Workload /DAY	Fraction RAC
fbval	va	2	55.00	55.00	0.030
fbupd	va	0	0.00	48.90	0.000
fbret	va	1	0.00	2.45	0.034
foret	ca	1	0.00	1.22	0.017
foupd	ca	0	0.00	38.85	0.000
foval	ca	2	45.00	45.00	0.148
fkval	tenn	4	0.00	24.50	0.084
recpt	tenn	0	80.00	80.00	0.000
shipp	ga	0	0.00	80.00	0.000
store	ga	1	0.00	80.00	0.000
fbpro	va	1	55.00	55.00	0.242
fopro	ca	1	45.00	45.00	0.445

Relative Allocation Costs (RAC):182,000 TRANSACTION VOLUME*MILES

= (TRANSACTION VOLUME * DISTANCE)

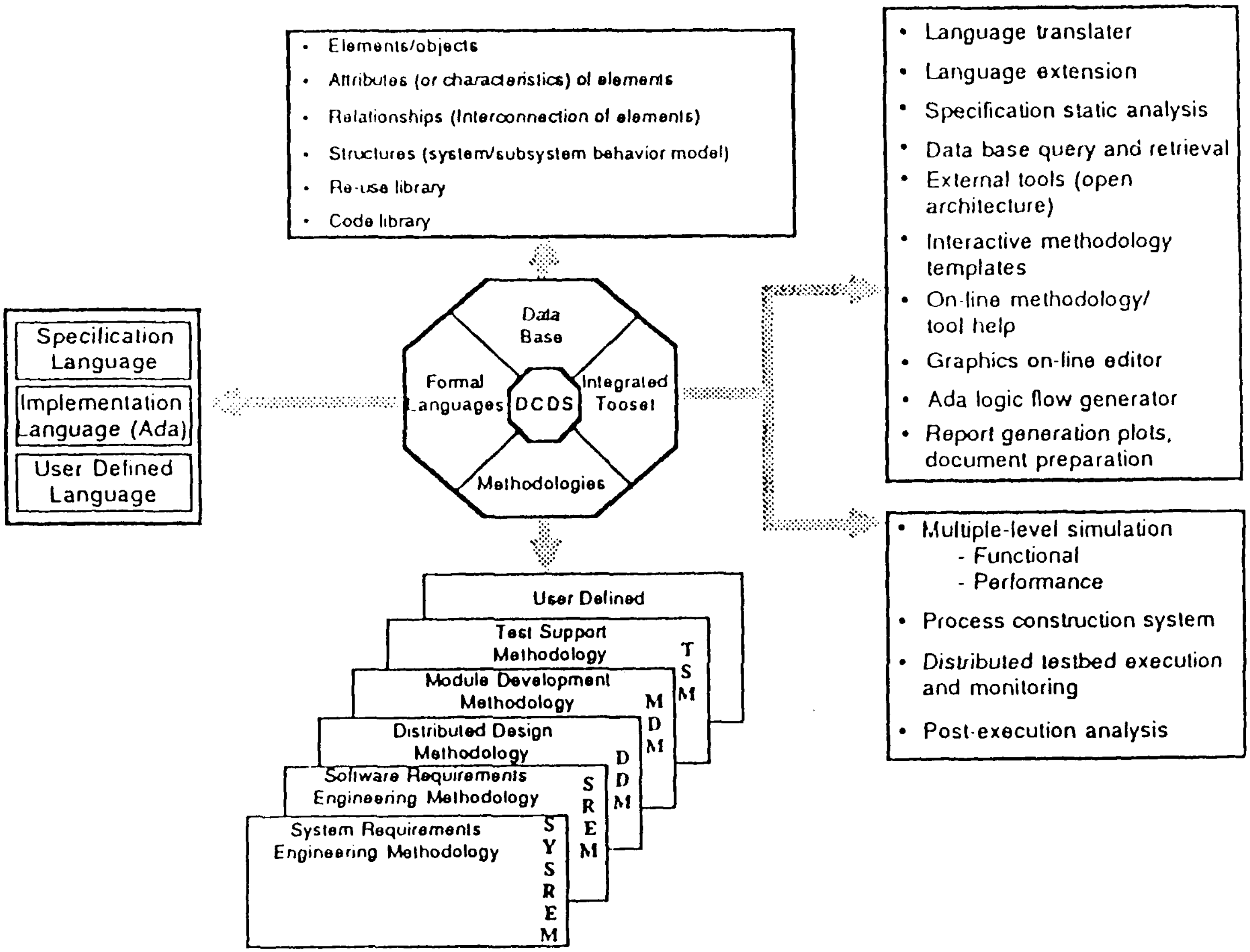
TRW Design Tool

The TRW Distributed Computer System Design (DCDS) has been an outcome of 13 years of efforts by the TRW and has been sponsored by the U.S. Army Strategic Defense Command at Huntsville, Alabama. The design tools seem to be reasonably mature even though the emphasis seems to be in the software design methodology area. We were not able to do enough experimentation because of the timing constraints. The non-availability of any systematic guideline or tutorial for using the software package delayed our understanding and assessment of the package. A thorough experimentation is desirable to appreciate the capabilities of the tool. This would eliminate any duplication of identical work. The DCDS consists of five different methodologies and are shown in Fig. 3. For each one of them, a separate but similar language has been used and is shown in Table III.

Table III Different Methodologies and Corresponding Languages

No.	Method	Language
1.	Systems Requirements Engineering Methodology (SYSREM)	System Specification Language (SSL)
2.	Software Requirements Engineering Methodology (SREM)	Requirements Statement Language (RSL)
3.	Distributed Design Methodology (DDM)	Distributed Design Language (DDL)
4.	Module Developmen Methodology (MDM)	Module Development Language (MDL)
5.	Test Support Methodology (TSM)	Test Support Language (TSL)

The selection of the design methodologies totally depends on the type of requirements given to the designer. Each of these methodologies have several phases and it is possible to see the relevance between different corresponding steps as well as various attributes. The specifications need to be defined by the designer and it seems to be more or less static in nature. Any changes in specifications necessitate rerun of the complete package. As we have not run an actual example, it is difficult to comment on the performance parameters to be produced by the DCDS. However, it seems that some attributes important to the Army such as actual Throughput, Node Utilization, Channel Utilization, Reliability and Graceful Degradation could be missing.



Fig, 3. The DCDS Components

Conclusions and Recommendations

The Distributed Computer System Design is still in the infancy stage and prolonged efforts are needed to gain adequate insight to various involved issues. To start with, it is important to have some simple design tools which could be easily migrated to and/or adopted at AIRMICS. With the experience I had at AIRMICS, it could be easily said that the existing McCabe Software package currently operational at AIRMICS, has very limited use and it is inappropriate for possible extensions both from the economical and the technical aspects. The DCDS software package from the TRW received towards the end of my stay at AIRMICS, look promising and further detailed experimentation is needed to evaluate and establish its capabilities and limitations. There are two major issues to be addressed. The first is the establishment of an appropriate hardware configuration so that the TRW DCDS software could be run successfully. This might require purchase and installation of additional equipment either at the AIRMICS or at the Georgia Tech Computing Center. The second issue is to deal with the source code for the DCDS software package. It is mandatory to have the source code if any modifications, changes or augmentations in existing DCDS software are to be done for further enhancements. Hence, efforts must also be made to acquire the source code either from the TRW or from the US ASDC. Additional efforts are needed to find out the design efforts being supported by other Government funding agencies as well as in-house research being carried out by various institutions and research organizations.

It seems to me that many design and development issues of current interest have either been partially covered or have not been addressed at all by the TRW software package while implementing the distributed system design. This may due to the fact that many newer techniques have now been introduced and due to advances in VLSI design technology, the issues associated with DCDS design, are different than the time the DCDS work at TRW started. A detailed study of other design efforts would also be helpful in identifying various issues and would provide an insight to several alterative strategies for handling the same problems. In any case, some of the obvious but nontrivial issues could be given as follows:

1. How do you test the adequacy of transformation from designers' to systems' requirements [ARM87]?
2. What partitioning strategy should be used in assigning the locations of various functional units and what functions to be assigned solely to the hardware?
3. Which network configurations are appropriate for Distributed System Design? Whether a simple ring structure is adequate or one needs to look into 2-in 2-out networks [CHU88] or other complex networks from the throughput and reliability viewpoint [KUM88]?
4. What file allocation scheme ought to be used from the reliability or graceful degradation view point [MAH88]?

5. What mechanism or protocol ought to be used to reduce the communication overhead [CHO88]?

6. What modifications/changes need to be done in existing modeling and simulation software? How do you check the correctness [ARM87]?

7. What type of prototype testbed is needed to validate the results?

It may be noted that this is not a comprehensive list. AIRMICS should provide substantial support in setting up some kind of a test bed for exhaustive experimentation and the aforementioned issues, in some prioritized manner, ought to be addressed immediately.

Finally, it would be useful to keep an eye on what other groups have been doing, as there is no point in reinventing the wheel. It would be adviseable to keep monitoring what different institutions are doing and then look in to appropriate portions/modules for future intregration.

REFERENCES

[AGR85] D.P. Agrawal, V.K. Janakiram and G.C. Pathak. "Evaluating performance of multicomputer configurations," IEEE Computer, Vol. 17, No. 3, March 1985, pp. 41-53.

[AGR88] D.P. Agrawal, W.E. Alexander, E.F. Gehringer, J. Mauney and T.K. Miller, "B-Hive: Hardware and Software for an Experimental Multiprocessor," Communicated for publication.

[ARM87] Army Science Board, "Information Management Concepts and Architecture," Final Report by Ad Hoc Panel, July 1987.

[BOE86] B.W. Boehm, "A Spiral Model of Software Development and Enhancement," ACM SIGSOFT Software Engineering Notes, Vol. 11, No. 4, Aug. 1986, pp. 21-46.

[CAS88] T.L. Casavant, "A Taxonomy of Scheduling in general-purpose Distributed Computing Systems," IEEE Trans. on Software Engineering, Vol. 14, No. 2, Feb. 1988, pp. 141-154.

[CHA80] G.A. Champine, R.D. Coop and R.C. Heinselman, "Distributed Computer Systems: Impact on Management, Design and Analysis," (Text Book), North Holland Publishing Co., 1980

[CHE80] P.P. Chen and J. Akoka, "Optimal Design of Distributed Information Systems," IEEE Trans. on Computers, Vol. C-29, No. 12, Dec. 1980, pp. 1068-1080.

[CHI88] G-M. Chiu and C.S. Raghavendra, "A Model for optimal Resource Allocation in Distributed Computing System," Proc. INFOCOM, 1988.

[CHO88] Y.K. Choe, D.P. Agrawal and C.R. Green, "A Hierarchical Message Mechanism for Distributed Systems Software," Proc. of the Workshop on the Future Trends of Distributed Computing in 1990's, Sept. 14-16, 1988, pp. 132-138.

[CHU88] T.Y. Chung, S. Rai and D.P. Agrawal, "Doubly connected multi-dimensional regular topologies for MANs & LANs," Proc. 1988 INFOCOM, March 1988, pp. 551-557.

[COR87] D.W. Cornell and P.S. Yu, "Relation Assignment in Distributed Processing Environment," Proc. 1987 Distributed Computing Conference, pp. 0-55.

[FOR86] I.R. Forman, "On the design of large Distributed Systems," Proc. 1986 Int. Conf. on Computer Languages, Miami, Fl, Oct. 27-30, 1986, pp. 84-95.

[GON80] M.J. Gonzalez and B.W. Jordan, "A Framework for the quantitive Evaluation of Distributed Computer Systems," IEEE Transactions on Computers, Vol. C-29, No. 12, Dec. 1980, pp. 1087-1094.

[GAI87] J. Gait, "Two tier scheduling in partitioned multiprocessors," Proc. 2nd Supercomputing Conference, 1987, pp. 179-186.

[HAN87] P.B. Hansen, "Joyce - A programming Language for Distributed Systems," Software - Practice and Experenice, Vol. 17(1), pp. 29-50, Jan. 1987.

[HOU87] C.E. Houstis, "Allocation of real-time applications to Distributed Systems," Proc. 1987 Int. Conf. on Parallel Processing, Aug. 17-21, 1987, pp. 863-866.

[IND86] B. Indurkhya, H.S. Stone and L. Xu-Cheng, "Optimal Partitioning of Randomely Generated Distributed Problems," IEEE Transactions on Software Engineering, Vol. SE-12, No. 3, March 1986, pp. 483-495.

[KAL87] L.V. Kale, "Comparing the performance of two dynamic load distribution methods," Tech Report, University of Illinois, UIUCDCS-R-87-1387.

[KUM88] A. Kumar, S. Rai and D.P. Agrawal, "Reliability evaluation agloritbms for Distributed Systems," Proc. 1988 INFOCOM, March 1988, pp. 851-860.

[LEU87] J-S. Leu, D.P. Agrawal and J. Mauney, "Modeling of Parallel Software for efficient computation-communication overlap," Proc. Fall Joint Computer Conference, 1987, pp. 569-575.

[MAH88] I.O. Mahgoub, A. Kumar, D.P. Agrawal and C.R. Green, "Network failure and graceful degradation in a class of cluster-based multiprocessors," communicated for publication.

[MCC87] McCabe & Associates, "Techniques and automated tools which support Distributed Computer System Design," Final Report for AIRMICS, Feb. 7, 1987, contract no. DAHC06-85-C-0001.

[PAT84] G. Pathak and D.P. Agrawal, "Task Division in Multicomputer Systems," Proc. 5th Distributed Computing Systems Conf., Denver, May 13-17, 1984, pp. 273-280.

[PRA85] D.K. Pradhan, "Fault-tolerant multiprocessor link and bus network Architectures," IEEE Trans. on Computers, Vol. 34, No. 1, Jan. 1985, pp. 33-45.

[RAD86] RADC in-house report, "Survey of Model/Simulation at RADC," TR-86-9, Mary L. Denz, 112 pages.

[RAM83] C.V. Ramamoorthy and B.W. Wah, "The isomorphism of simple file Allocation," IEEE Trans. on Computers, Vol. C-32, No. 3, March 1983, pp. 221-232.

[RAM85] C.V. Ramamoorthy, V. Garg and R. Aggrawal, "Environment Modeling and Activity Management in GENESIS," SoftFair II: Second Conference on Software Development Tools, Techniques and Alternatives, December 1985.

[RAV87] C.P. Ravikmar and L.M. Patnaik, "An Architecture for CSP and its simulation," Proc. 1987 Int. Conf. on Parallel Processing, Aug. 17-21, 1987, pp. 874-881.

[SHA87] S.M. Shatz and J-P. Wang, "Introduction to Distributed-Software Engineering," IEEE Computer, October 1987, pp. 23-31.

[STA84] J.A. Stankovic, "A perspective on Distributed Computer Systems," IEEE Transactions on Computers, Vol. C-33, No. 12, December 1984, pp. 1102-1115.

[TAN85] A.N. Tantawi and D. Towsley, "Optimal static load balacing in distributed Computer Systems," Journal of the ACM, Vol. 32, No. 2, April 1985, pp. 445-465.

[TRW87] TRW System Development Division, "Distributed Computing Design System (DCDS)," reports submitted to U.S. Army Strategic Defense Command, Huntsville, Al, CDRL A008, October 1987.

[WAN85] Y-T. Wang and R.J. Morris, "Load Sharing in Distributed Systems," IEEE Transactions on Computers, Vol. C-34, No. 3, March 1985, pp. 204-217.

[YAS87] M. Yasrebi, J.C. Browne and D.P. Agrawal, "A representation basis for the expression of communication times of algorithms expressed in terms of Computation Graphs," Proc. COMP-EURO '87, VLSI and Computers, Hamburg, West Germany, May 11-15, 1987.